Probabilistic Programming: Concepts and Challenges (Extended Abstract)

Angelika Kimmig and Luc De Raedt
KU Leuven, Belgium

A multitude of different probabilistic programming languages exists today, all extending a traditional programming language with primitives to support modeling of and reasoning with complex, structured probability distributions. Examples include functional languages (Church [Goodman et al., 2008], IBAL [Pfeffer, 2001]), object-oriented languages (Figaro [Pfeffer, 2009]), and logic languages (ProbLog [De Raedt et al., 2007], PRISM [Sato and Kameya, 2001]). Each of these languages employs its own probabilistic primitives, and comes with a particular syntax, semantics and inference procedure. This makes it hard to understand the underlying programming concepts, to appreciate the differences between the different languages, and to relate them to work in related fields such as statistical relational learning or probabilistic databases.

To obtain a better understanding of probabilistic programming, we propose to use the notion of *probabilistic programming concepts*, that is, common notions underlying the primitives used by various probabilistic languages. In a recent survey [De Raedt and Kimmig, 2013], we identify a number of such core programming concepts, discuss the execution mechanisms that they require and use these to position state-of-the-art probabilistic languages and their implementation. While doing so, we focus on probabilistic extensions of *logic* programming languages such as Prolog, which have been developed since more than 20 years.

These core concepts include probabilistic ones such as different types of random variables (binary, discrete, continuous) or stochastic memoization (whether or not repeated occurrences of the same random variable share their value), as well as concepts originating in programming languages such as second order predicates and meta-calls, and general modeling concepts such as constraints or time and dynamics.

Inference is a key challenge in probabilistic programming and statistical relational learning. Furthermore, the choice of inference approach often influences which probabilistic primitives can be supported. Enormous progress has been made in the past few years w.r.t. probabilistic inference and numerous inference procedures have been contributed. Therefore, we also identify some core classes of inference mechanisms for probabilistic programming and discuss which ones to use for which probabilistic concept.

This work also reveals and summarizes a number of challenges and directions for future work. Efficient inference approaches for languages supporting a broad range of concepts are needed. Promising directions include lifted inference, as studied mainly in the statistical relational learning community so far, and MCMC approaches, which show promise in probabilistic programming

languages, but typically still require proposal functions to be custom made for the program at hand. Another question that has only seen partial answers so far is how to efficiently deal with evidence and constraints in different inference techniques. Adapting and extending program transformation and analysis techniques to the probabilistic setting promises opportunities to recognize and exploit program parts that are amenable to more efficient inference. Concepts such as time and dynamics require inference approaches that on the one hand exploit repeated structure, but on the other hand can also deal with changing structure over time.

To summarize, by identifying probabilistic programming and modeling concepts and relating them to different inference approaches, we provide a framework to establish connections between expressive probabilistic languages developed in different communities, and to facilitate exchange of ideas and techniques between them.

References

L. De Raedt, A. Kimmig, and H. Toivonen. ProbLog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 2007.

Luc De Raedt and Angelika Kimmig. Probabilistic programming concepts. *CoRR*, arXiv:1312.4328, 2013.

N. Goodman, V. K. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum. Church: a language for generative models. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)*, 2008.

A. Pfeffer. IBAL: A probabilistic rational programming language. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, 2001.

Avi Pfeffer. Figaro: An object-oriented probabilistic programming language. Technical report, Charles River Analytics, 2009.

T. Sato and Y. Kameya. Parameter learning of logic programs for symbolic-statistical modeling. *J. Artif. Intell. Res. (JAIR)*, 15:391–454, 2001.